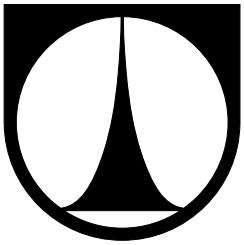
TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií



Automaticky řízené elektrické auto

Bakalářská práce

Liberec 2012 Ota Šverma

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2646 – Informační technologie

Obor: 1802R007 – Informační technologie

Automaticky řízené elektrické auto

Automatic controlled slot car

Bakalářská práce

Autor práce: **Ota Šverma**

Vedoucí práce: Ing. Jan Koprnický, Ph.D.

Konzultant práce: –

V Liberci 21. května 2012

Originální zadání

Originální zadání str 2

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/ 2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum

Podpis

Poděkování

Tímto bych rád poděkoval svému vedoucímu bakalářské práce Ing. Janu Koprnickému za umožnění příležitosti spojit práci s velkým množstvím nových zkušeností, s možností zúčastnit se soutěže Freescale Race Challange, za podporu a hlavně trpělivost během samotného vývoje programu.

Velký dík patří také mému kamarádovi Bc. Jakubovi Vítovi za cenné rady, které byly v mnohých případech nedocenitelné.

Abstrakt

Tato bakalářská práce popisuje jednotlivé komponenty použité v automaticky řízeném elektrickém autě a způsob jakým bylo naprogramováno. Vysvětluje využití mikroprocesoru MCF51JM64, použitého jako hlavního řídícího prvku, který dále pracuje s H-můstkem ovládajícím elektromotor v modelu auta a s daty z čidel. Také uvádí použití filtrovaných dat z tříosého akcelerometru MMA7361L, ukládání na paměťovou kartu, zapojení reflexního snímače CNY70 a výrobu vlastního optického čidla. Dále se zabývá softwarem, analýzou a využití filtru pro zpracovaní dat.

Hlavní náplní je naprogramování algoritmu, který optimálně namapuje neznámou závodní trať a poté ji projede co nejrychleji.

Klíčová slova: Mikroprocesor, Akcelerometr, H-můstek, Freescale, Filtr

Abstract

This thesis describes particular components and the software development of automatically controlled slot car. It explains usage of microprocessor MCF51JM64 as a main controlling unit, connected to H-bridge which regulates the speed of the electric engine inside the model and working with data from various sensors. It also describes filtered data from three-dimensional accelerometer MMA7361L, saving to flash memory, connection of reflexive sensor CNY70 and making of own optical sensor. The thesis also comments on software and the analysis and usage of data filter.

The main topic of the thesis is development of an algorythm, which optimally maps unknown race track and then controls the car trough the course as fast as possible.

Key words: Microprocessor, Accelerometer, H-bridge, Freescale, Filter

Obsah

Seznam obrázků

1 Úvod

Cílem této bakalářské práce je seznámit se s elektronikou poskytnutou společnosti Freescale Semiconductor, která je navržena tak, aby bylo možné její osazení v modelu auta na autodráhu. Poté vše zkompletovat uvnitř vozu a zapojit na původní součásti vozu. Seznámit se s jednotlivými komponenty a pochopit jejich funkčnost pro automaticky řízené vozidlo.

Práce bude nejprve zaměřena na obecné informace popisující soutěž, které se auto zúčastní a její pravidla. Probrány budou jednotlivé součástky na desce plošných spojů, software potřebný k vývoji a analýze získaných dat. Dále je popsána problematika jednotlivých softwarových prvků naprogramovaného algoritmu. Budou vysvětleny jednotlivé režimy využité pro mapování a závod. Nedílnou součástí je také popis přidaného hardwaru, jeho jednotlivé problémy a popřípadě návrhy na možné zlepšení. V kapitolách popisující řízení auta jsou popsány problémy vznikající během samotného závodu. Kapitoly zabývající se softwarem osvětlují nedostatky použitých aplikací a také jejich možné alternativy na systémech unixového typu.

Hlavním cílem je naprogramovat automaticky řízené auto tak, aby optimálně a zároveň nejrychleji projelo dráhou bez výpadku z trati a mohlo se zúčastnit turnaje Freescale Race Challenge, kde bude závodit na čas proti dalším autům na předem neznámém tvaru autodráhy.

2 Freescale Race Challenge

2.1 O soutěži

Soutěž Freescale Race Challenge je pořádána společností Freescale Semiconductor ČR pro celkem sedm elektrotechnických vysokých škol, kde právě pět škol je z České republiky a dvě ze Slovenska.

Hlavním předmětem této soutěže je malé programovatelné auto osazené 32-bitovým procesorem Freescale ColdFire V1, které má za úkol pomocí dostupných prvků co nejrychleji namapovat tvar neznámé autodráhy a pokud je možné, být co nejrychlejší. Pokud by auto vypadlo z dráhy, mělo by si pamatovat svojí poslední pozici tak, aby nebylo již nutné nové mapování.

Auto je napájeno stabilním napětím z laboratorního zdroje 15 volty a má k dispozici akcelerometr, s jehož pomocí má měřit přetížení v zatáčkách a zaručit plynulou jízdu na trati.

Organizace letošního ročníku 2012 byla pozměněna, vyřazovací turnaj je organizován jednotlivými vysokými školami. Školy tak budou mít více prostředků k uzpůsobení svým potřebám a po vyřazovacím kole se sejdou finalisté na velkém finále v Rožnově pod Radhoštěm, které bude součástí Freescale Technology Day.

Díky velkému zájmu použít kamery v autě byl v pravidlech soutěže zvýšen hmotnostní limit o 20 gramů hmotnosti pro auta s kamerou, ale původní hmotnost 125 gramů je nadále stejná.

2.2 Pravidla hlavního závodu

- „Každé auto závodí samostatně na čas.“
- „Měření času začíná prvním průjezdem časomírou a končí po 10 kolech.“
- „Pokud auto vypadne z dráhy, soutěžící jej smí znovu nasadit.“
- „Závod se jede na 2 jízdy, v pravé a v levé dráze, součet časů obou jízd určuje výsledné pořadí.“

2.3 Pravidla vyřazovacího turnaje

- „Dvojice aut spolu závodí ve stíhacích jízdách na 8 kol.“
- „Závod se odstartuje zapnutím napájení.“
- „Auto, které první dokončí 8 kol, postupuje.“
- „Pokud se auta srazí na křížení drah, auto, které vedlo, vyhrává.“
- „Tvar dráhy je známý.“

3 Osazení plošného spoje

Deska plošných spojů (dále jen DPS) byla navržena tak, aby rozměry umožňovaly její vložení do modelu auta Audi R8 a rozložení konektorů odpovídalo ideálnímu připojení původních prvků auta.

- V první fázi byly dodány jednotlivé komponenty zvlášť a k jejich kompletaci je potřeba speciální pájecí technika.
- V druhé již deska byla osazena a připravena k osazení do modelu auta.

Samotné připojení DPS není již nikterak náročné, stačí pouze odpojit původní desku a připájet novou podle literatury viz [7].

3.1 Mikroprocesor

Použitý mikroprocesor, typ MCF51JM64 z dílen pořadatele soutěže, je postaven na jádře V1 ColdFire a je určen do míst kde se klade nárok na nízkou spotřebu.

Procesor může pracovat až na 50,33 MHz a ostatní komponenty vždy pracují na polovině jeho nominální frekvence. Skládá se z mnoha dílů ale pro naprogramovaní auta není potřebné znát všechny jeho komponenty ale pouze díly s kterými pracujeme (viz tab. 1/s. 16) .

Část flashové paměti procesoru je předprogramovaná bootloaderem, který detekuje, jakým způsobem je k autu připojen elektrický zdroj. Pokud je připojen zdroj přes USB port, funguje zbytek paměti jako disková jednotka. Naopak je-li procesor napájen z autodráhy, bootloader spustí program určený k vykonání na zbylé paměti.

3.2 Akcelerometr

Akcelerometr MMA7361 je opět z vlastní výroby společnosti Freescale Semiconductor. Tato miniaturní součástka slouží k měření přetížení na všech třech osách a to za pomoci malých mechanických vahadel o jejichž poloze podává aktuální analogové informace na jednotlivých výstupech. Dokáže si také na základě teplot, které by díky rozpínání materiálu použitého na vahadlech mohly ovlivnit samotné měření provézt kalibraci.

Akcelerometry mají velice všestranné využití ve všech různých typech zařízení od mobilních telefonů, laptopů až po crash testové figuríny používané v automobilovém průmyslu.

Tabulka 1: Důležité komponenty procesoru

Komponenta	Popis
Flash paměť	Poskytuje paměť pro zdrojový kód
RAM (random-access memory)	Úložný prostor pro data
GPIO (general-purpose input/output)	Umožňuje přístup k I/O portům procesoru
ADC (analog-to-dialog converter)	Měří analogově napětí až 12 bity
KBI (keyboard interrupt)	Zprostředkuje přerušení na pinech

3.3 H-můstek

H-můstek(H-Bridge) umožňuje ovládat motorek napětím nebo proudem nezávislém na DPS. Jeho základ tvoří čtyři spínače, které se dají ovládat naším mikroprocesorem, díky nimž můžeme měnit například směr rotace motoru. Změna směru rotace motoru se dá například použít k aktivnímu brzdění, což nám umožní rychlejší deceleraci auta. Z H-můstku můžeme číst informace o stavu napětí, díky kterému můžeme určit kdy auto přejede křížení, čili stav kdy je DPS napájena pouze z kondenzátoru ale zbytek je mimo napájení dráhy.

V autě je použit H-můstek MC33931. Jeho výstup může být zatížen dlouhodobě od 8 V do 28 V(krátkodobě 5 V až 40 V) a pulsně šířkovou modulací[1] do 11 KHz. Je provozuschopný při teplotách v rozmezí od minus 40 °C až do 125 °C díky svému masivnímu pouzdru umožňující lepší chlazení v náročnějších podmínkách.

Motor je připojen na výstupní piny OUT1 a OUT2 , které jsou ovládány piny IN1, IN2 a D1. Piny IN1, IN2 a D1 jsou logické vstupy kde LOG1 = 3 V nebo 5 V. Pin D1 vypíná výstup motoru (LOG1 = vypnuto) ale neovlivňuje zbytek integrovaného obvodu. Kombinací IN1 a IN2 určujeme směr otáčení motoru.

3.4 Čtečka SD karet

V loňském a tomto ročníku byl plošný spoj vylepšen o čtečkou microSD karet, což značně zlepšuje situaci s nedostatkem paměti. Paměťová karta se může využít např. k ukládaní dat

[1] PWM - pulse width modulation

z akcelerometru nebo informací o dráze.

Ukázkový zdrojový kód ukládá data do souboru typu CSV [2], v našem případě oddělené středníkem v souboru jménem xxxxxxxx.CSV, kde x znázorňuje pořadí, ve kterém byl vytvořen. Stejně jako miniUSB konektor má i čtečka přístupový otvor v karoserii automobilu ale na opačná straně modelu.

3.5 Krystal

Krystal slouží k udávání vnějšího hodinového taktu. Je to elektrický oscilátor, který využívá mechanické rezonance a vibrující krystal k vytvoření elektrického signálu při určité frekvenci. Tato frekvence určuje takt integrovaných obvodů.

Oscilátor by se měl chovat co nejpřesněji, aby se nezměnil čas jednotlivých period, mohlo by pak dojít ke zkreslení času nebo dat. Také určuje frekvenci řízení motoru pomocí PWM. Naše deska je osazena 8 MHz krystalem od firmy Quartz v pouzdře SMD.

3.6 Diodový můstek

Jako diodový můstek (Bridge Rectifier) je použit DB106S v plastovém pouzdře a jeho maximální zátěž může být až 1 A. Můstek zaručuje stejnou polaritu napětí na výstupu i při změně polarity napětí na vstupu. Ošetření vstupního napájení za pomoci tohoto můstku nám zaručí, že do DPS půjde vždy stejná polarita, ať dáme auto na dráze v jednom nebo druhém směru. Tím nedojde k poškození jednotlivých součástek, které by při přepólování mohly být zničeny.

3.7 Regulátor napětí

Stabilizace napětí tvoří velice důležitý prvek, který reguluje napětí pro celou DPS a dalších přidaných komponent. Regulátor LP2950 je typu low drop tzn., že dokáže pracovat při malých rozdílech (50 mV a odběru 100 μA nebo při 380 mV a odběru 100 mA) mezi vstupním napětím, které je až do 29 V a výstupními 3,3 volty. Obyčejné regulátory pro svojí funkci potřebují rozdíl napětí několika voltů, než začnou pracovat. Do obvodu s regulátorem je zapotřebí připojit kondenzátor pro vyhlazení výstupního napětí.

[2] Comma-separated values - hodnoty oddělené čárkami

4 Softwarová výbava

4.1 Použitý programovací jazyk

K programovaní využijeme programovací jazyk C. Tento jazyk má výhodu ve své přehlednosti oproti assembleru, není sice tak rychlý po překladu (v současné době se mikroprocesory optimalizují pro jazyk C, což umožňuje jeho rychlejší zpracovaní) jako assembler, ve kterém pouze symbolicky píšeme instrukce pro mikroprocesor ale při rychlosti 48 MHz, je to otázka tak krátkých okamžiků, že není důvod k obavám ze ztráty výkonu.

C je po assembleru jeden z nejrychlejších jazyků využívaný pro hardwarovou vrstvu. I při svém stáří je stále vyvíjen a hojně používán.

4.2 Nahrávání aplikací

Mikroprocesor je předem vybaven bootloaderem, což uživateli usnadňuje nahrávaní aplikace. Po připojení přes miniUSB konektor se paměť auta hlásí jako Mass Storage[3], neboli jako klasická flash paměť a není nutno dalších ovladačů. Toto řešení je optimální, pokud bychom chtěli provádět vývoj na jiném OS[4] než MS[5] Windows. Jediný obsah paměti po připojeni tvoří textový soubor README.txt, který je ovšem prázdný. Po nahrání aplikace (soubor s příponou .S19 ve složce bin v projektu vytvořeném prostředím CodeWarrior) bootloader naprogramuje vnitřní flash paměť, která má 64 KB. Na automobilu se rozsvítí všechna světla v případě, že došlo k správnému naprogramovaní vnitřní flash paměti.

Problém může nastat na některých Linuxových distribucích, kde se auto místo jednou nahlásí vícekrát a se špatnou informací o velikosti vnitřní paměti. V důsledku to však nemá žádný negativní vliv na nahrání aplikace.

4.3 Vývojové prostředí

Pro vývoj bylo použito prostředí CodeWarrior poskytované společností Freescale. CodeWarrior nejenže zvýrazňuje syntaxi jazyka ale má i potřebné nástroje pro překlad zdrojových kódů. Bohužel není úplně zdarma, musíme tedy použít zkušební verzi ale i tak pro

[3] Počítačový protokol pro komunikaci s externími zařízeními

[4] OS = operační systém

[5] MS = Microsoft

náš účel plně postačuje jelikož dovoluje překlad maximálně 64 KB zdrojových kódů, což je stejné jako velikost vnitřní paměti použitého mikroprocesoru.

Prostředí není nijak náročné a jeho chod je rychlý. Za pomoci klávesové zkratky *alt+.* dokáže doplňovat syntaxi jazyka a proměnných. Za jisté negativum můžeme považovat absenci některých klávesových zkratek jako v prostředích Eclipse nebo Netbeans. Dále není optimální práce s okny, která reprezentují jednotlivé otevřené soubory se zdrojovými kódy. Moderní prostředí řeší tento problém pomoci přepínatelných záložek, jenž usnadňují programovaní a zvyšují přehlednost.

5 Software pro zpracovaní dat

5.1 MS Windows

5.1.1 MS Office

Z balíku kancelářských aplikací lze využít MS Excel, který umožňuje import dat ze souboru CSV a následné použití grafů pro zobrazení informací uložených při mapování autodráhy. Je zde široká škála možností jak tato data vyobrazit jelikož MS Excel má silnou podporu pro práci s grafy. Ve výsledném zobrazování dat však nebyl použit pro svojí cenu a omezení použití pouze pro MS Windows.

Můžeme však využít možnosti emulovat prostředí pro MS Excel v Linuxu pomocí známé aplikace Wine[6] ale toto nám nemusí zaručit jeho 100% funkčnost. Také zprovoznění může zabrat značné množství času.

5.1.2 Matlab

Matlab je profesionální nástroj pro zpracovaní obrazových, signálových a audio dat. Má přímou podporu CSV souborů a množství funkcí pro práci s grafy... Umožňuje psát vlastní scripty; pokud je umíme tvořit, dokáží velice usnadnit opakující se činnost což je v našem případě velké množství dat zaznamenaných z jednotlivých jízd auta na autodráze.

Velké plus nám poskytl předmět MTLB na oboru IT TU v Liberci. Tento předmět položil základ syntaxe příkazů a tvoření scriptů. Je to velice dobrý zdroj pro čerpání informací při zpracovaní dat z autodráhy a veliká časová úspora (než bychom se v tomto programu naučili pracovat sami by stálo velké úsilí, tato můžeme ušetřený čas využít pro vývoj).
Matlab je verzích jak pro MS Windows tak i pro Linux.

Nevýhodou může být jeho náročnost na úložný prostor potřebný k jeho instalaci v porovnání s tím, jak málo po něm chceme při této práci. Ale hlavním důvodem, proč nebyl použit je jeho cena. Jde opět o placený software, který si běžný student nemůže dovolit, musíme se tady uchýlit k freewarovým alternativám.

[6] Wine = Wine is not emulator

5.2 GNU Linux

V současné době se tento dříve neobvyklý operační systém běžně rozšiřuje i na klientské stanice. Je tedy dobré se zmínit o alternativních nástrojích, které mohou pomoci při analýze nasbíraných dat. Dále popsané nástroje pracují jak pod systémy na bázi Unixu tak i Windows.

5.2.1 LibreOffice

Tento balíček nástrojů pro práci s dokumenty je alternativou k MS Office. Z jeho nabídky lze použít nástroj pro práci s tabulkami jménem Calc. Pro vizualizaci dat má podobné možnosti jako jeho protějšek, tudíž jeho podrobný popis není důležitý.

Důležité je zmínit jeho velkou nevýhodu. Při vkládání velkého množství dat dochází k obrovským časovým ztrátám během jejich zpracování a tudíž není vhodný k častým analýzám.

5.2.2 GNU Octave

Octave je konzolová open source alternativa k Matlabu. Její vývojáři se snaží dodržet syntaxi Matlabu, není tedy nutné učit se nové věci a lze využít znalosti z průběhu studia.

Dále můžeme shledávat konzolové aplikace jako nepřehledné nebo náročné na ovládání. Proto má Octave grafickou nadstavbu jménem QT Octave, vyžívající QT knihovny, podobnou prostředí Matlabu.

Tento nástroj je multiplatformní a pro použití v grafickém rozhraní na MS Windows stačí nahrát knihovny nutné pro běh aplikace. Pro zpracovaní dat postačí nastavit dělící znak a načíst CSV soubor do matice.

Graf se vykreslí pomocí příkazů *figure,subplot* a *plot* stejně jako je tomu u Matlabu. K popisu grafu slouží příkazy *title,zlabel,ylabel*.

6 Dodatečná výbava auta

6.1 Snímač otáček kola

6.1.1 Použité součástky

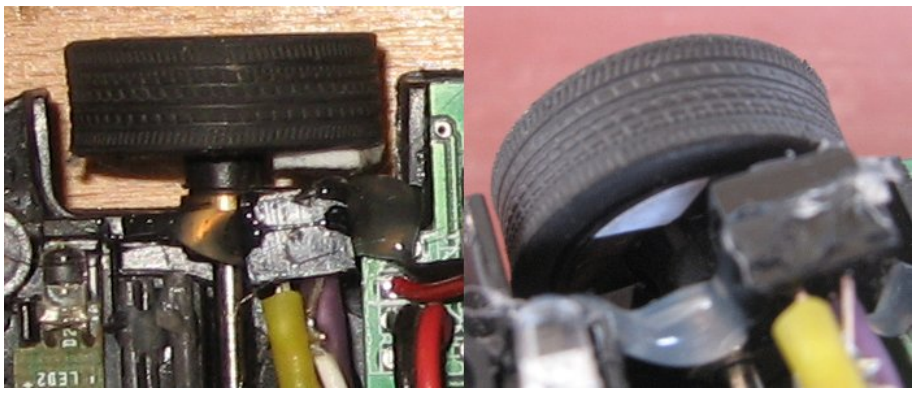
Základem je optická závora CNY70 složená z infra diody a fotocitlivého tranzistoru v plastovém pouzdře vybaveném optickým filtrem. Závora měří až do vzdálenosti 2.5 mm.

Před diodu je předřazený rezistor omezující maximální proud, který ji chrání před poškozením nadměrným proudem.

Signál poskytnutý fototranzistorem v čidle je závislý na množství dopadajícího odraženého světla z infra diody. Změna na výstupu fototranzistoru je tedy plynulá a procesor by jí musel načítat pomocí AD převodníku, kde by se dále musel softwarově zpracovat. Proto je v obvodu snímače zahrnut obvod 4010, obsahující invertor, který signál přetvoří na log 0/1.

Hradlo určuje rozdíl mezi log 0/1 podle poloviny napětí na napájecích pinech Vcc a Vss. Pomocí invertoru tedy získáme vyhlazený signál a stačí nám pouze vyvolávat přerušení, které poté zpracujeme ve zdrojovém kódu.

Celý snímač byl připájen na kousek univerzálního plošného spoje. Jedná se zejména o připájení patice pro invertor, umístění odporů a spojení optického čidla pomocí vodičů. Z důvodu nedostatečné pevnosti konstrukce bez plošného spoje bylo rozhodnuto ve prospěch konstrukce na DPS.



Obrázek 1: Snímač otáček kola.

6.1.2 Realizace v autě

Při praktické instalaci tohoto snímače do modelu je nutno vyrobit reflexní plochu kruhového tvaru s možností upevnění podle niž by mohl reagovat na otáčky kola. Měla by obsahovat co nejvíce přechodů mezi plochami na které je čidlo citlivé. Docílí se tak větší přesnosti měření.

Během této práce byla vytvořena plocha se čtyřmi reflexními částmi, (viz obr. 8/s. 43) kde každá znamená ujetí 1,57 cm na trati. Celkový obvod kola je tedy 6,28 cm.

Vhodné umístění je i přes jisté mechanické nedostatky na jednom z předních kol. Zadní kola mohou podlehnout ztrátě adheze při velké odstředivé síle v zatáčkách, prokluzu během akcelerace nebo špatné adhezi po dobu brzdění.

Každý impulz od snímače je vnímán jako KBI přerušení a inkrementuje vnitřní proměnnou vypovídající o naměřené délce.

6.1.3 Nedostatky snímače

Reálné použití je pouze na rovinách, kde nedochází k příčení kol vůči směru, kterým je auto vedeno vodítkem.

V zatáčkách dochází k neovlivnitelné blokaci předních kol, což způsobuje špatnou detekci ujeté vzdálenosti. Také korektnost naměřené vzdálenosti se zhoršuje při vyšších rychlostech během průjezdu zatáčkou. Není tedy vhodné postavit celý program závislý na datech ze snímaní otáček kola z důvodu vysoké nepřesnosti. Naopak při vývoji poslední verze snímač téměř nebyl použit s výjimkou pasivního brzdění na rovinách.

Dále jsou měření ovlivňována nečistotami omezujícími přilnavost předních kol a příliš napružené sběrací kartáčky zvedající auto do výše.

Vhodným řešením je vyhnout se realizaci takovéhoto snímače a použít raději data z osy ypsilon poskytované akcelerometrem k měření ujeté vzdálenosti. Vyhneme se tak fyzikálním a mechanickým nedostatkům přední nápravy auta.

6.2 Snímání brány

Snímač brány je složen ze dvou optických čidel vlastní konstrukce, podobných senzoru, který byl použit při snímaní otáček kola ale s mnohem větším dosahem. Když uvážíme

vzdálenost sloupku měřící brány od boku auta je třeba podstatně vyšší výkon infra diody aby došlo k odrazu světla zpět do fototranzistoru.

Na tuto úlohu byla vybrána led dioda IRS-5, která vyzařuje ve spektru 880 nm v sérii s rezistorem 37 ohmů aby nedošlo k jejímu proudovému přetížení. Hodnota rezistoru byla zvolena podle Ohmova zákona dle napětí poskytovaného autem a maximálního proudu diody. Při menším odporu docházelo postupnému přetížení čipu diody a následkem toho ke slábnutí svitu. Naopak větší odpor snižoval výkon diody a dosah celého čidla. Obě čidla byla připojena k NAND hradlu tak, aby během detekce obou stran zároveň byla na výstupu log 1.

Množství faktorů ovlivňujících správnost měření je značné. Vhledem k tomu, že čidlo není vybaveno optickým filtrem je jeden z hlavních problémů svit zářivek, které jsou modulovány frekvencí rozvodné sítě a svítí mimo jiné také ve spektru snímaném použitým tranzistorem. Pro omezení tohoto rušivého jevu jsou použita dvě čidla, která musí snímat oba sloupky časomíry zároveň, nedojde tedy ke zmatení auta v případě, že by jedno z čidel osvítila zářivka nebo jiný nežádoucí zdroj. Snížit riziko je také možné modulací LED diod a snímat příslušnou frekvenci tranzistorem.

Čidlo by šlo vylepšit tranzistorem, který sám demoduluje frekvenci na kterou reaguje od frekvenčně modulované LED diody. Tímto způsobem lze také měřit vzdálenost snímaného předmětu a odlišit případný šum. Pouze v tomto zapojení je zapotřebí LED diody zvládající vysoké frekvence potřebné k modulaci.

Obrázek 2: Čidlo pro snímaní brány.

Celkově je toto řešení energeticky náročné, proto bylo původní hradlo odstraněno z návrhu a čidla připojena ke stávajícímu invertoru. Problém nastával během výpadku proudu, kdy kondenzátor neměl dostatečnou kapacitu pro udržení minimálního napájecího napětí mikroprocesoru a docházelo k resetování programu. Výstupy invertoru byly připojeny ke GPIO portům a porovnávání zda-li jsou snímány oba sloupky bylo nadále řešeno softwarově.

7 Program auta

Tato kapitola se bude zabývat jednotlivými prvku softwaru auta, které jsou nadále použity pro mapovací a závodní režim.

7.1 Určení jednoho kola

Před začátkem mapování je důležité zvolit, jakým způsobem budeme určovat jedno ujeté kolo trati podle kterého se vytvoří mapa pro závodní režim a nadále se podle počátku kola bude auto synchronizovat.

Možností se nabízí hned několik. Bezesporu tou nejjednodušší bez dalšího hardwaru je snímaní překřížení probrané v kapitole 7.2. Dále se nabízí snímání bílé čáry popsané v literatuře [9], která se nachází na startu a může ušetřit čas než nalezneme první překřížení na dráze. Také lze snímat bránu pomocí optických čidel uvedených v kapitole 6.2. Dále se nabízí možnost využití ultrazvuku. V posledním případe lze zakoupit hotový modul ale oproti ostatním možnostem je toto komplexní řešení několikanásobně dražší.

Každá z těchto možností určení ujetého kola může sloužit k synchronizaci auta na dráze. Je vhodné s nimi počítat jak pro určení kola tak pro synchronizaci a docílit tak co nejpřesnějšího určení polohy.

7.2 Snímaní překřížení

Dle pravidel soutěže jsou na dráze použita dvě překřížení a můžeme s nimi počítat jako s prvky, které můžeme snímat a zahrnout do návrhu algoritmu.

Překřížení je na dráze místo, kde napájecí kolejnice není souvislá, dojde tedy k dočasnému přerušení kontaktu mezi dráhou a kartáčky vozu. V momentě, kdy dojde k výpadku, je DPS napájena z kondenzátoru a motor je bez elektrického proudu, tento stav lze zjistit čtením informací z H-můstku a tuto událost lze snímat během krátkých přerušení mikroprocesoru.

Vzhledem k rychlosti procesoru a délky výpadku napětí je možné během jednoho průjezdu nasnímat jedno překřížení vícekrát. Může tedy dojít ke zkreslení počtu překřížení a je nutné přebytečné hodnoty filtrovat do té doby, než na motoru bude opět napětí.

Počátek kola začíná detekcí prvního překřížení a lze tedy započít mapování trati. Jedno kolo končí tehdy když projedeme původním překřížením.

Než se auto rozjede není ještě motor signálem od procesoru do H-můstku spuštěn. Z pohledu zdrojového kódu se jedná o další přerušení ale z pohledu měření jde o chybu. Musíme tedy zabránit snímání v momentě, kdy auto ještě stoji na počátku. Problém vyřeší pomocná proměnná indikující změnu pohybového stavu auta. V případě, že nebude během probíhajícího programu vypínán motor lze jako indikátor brát hodnotu na pinu H-můstku, udávajícím informaci o stavu motoru.

Nevýhodou způsobu měření ujetého kola touto metodou je značná prodleva před tím, než nalezneme první překřížení na trati od kterého mapujeme. Pakliže by bylo překřížení v blízkosti startu není to příliš velká ztráta ale pokud protivník použil jiný způsob určení kola, získává v tomto směru značný náskok.

Další problém může být v kartáčcích zajišťujících kontakt s dráhou. Mohou být opotřebené, nemusí mít správný přítlak k napájecím kolejnicím... Tím pádem snímáme falešná překřížení a dochází k předčasnému ukončení kola.

7.2.1 1. verze

Jednou z možností je vytvořit filtr, který bude reagovat na náběžnou hranu (moment, kdy dojde k přerušení napájení), zbylé hodnoty zahodí a na sestupné hraně se uvede do původního stavu (moment, kdy se obnoví napájení). Každým průjezdem za použití filtru inkrementuje vnitřní proměnou, která nás informuje o počtu takto nastalých situací.

Během mapovacího režimu tento způsob pracuje adekvátně ale jakmile auto začne zrychlovat příliš prudce nebo jede v zatáčkách tak rychle, až vznikají smyky, dochází ke krátkodobým výpadkům a zmatení celého algoritmu. Tento filtr s těmito problémy nekalkuluje, detekuje tedy i falešná překřížení a špatně inkrementuje pomocnou proměnou. Je tedy nevhodný pro aplikaci do auta. Řešením těchto chyb je druhá verze funkce pro snímaní překřížení.

7.2.2 2. verze

Falešná překřížení jsou pouze krátkodobého rázu oproti pravému překřížení, které trvá v řádu desítek milisekund. Trvání průjezdu lze tedy použít jako opěrný bod k rozpoznání chybového stavu.

Samotná detekce probíhá prvním naměřeným výpadkem, kde si program v tento moment uloží čas počátku, funkce se poté překlopí do stavu kde další měřené hodnoty nejsou považovány za důležité a čeká až opět bude na motoru napětí. Pokud se napětí obnoví, dostane se funkce do původního stavu s tím rozdílem, že uloží čas mezi počátkem a koncem měření.

Hlavní částí funkce je, že během zpětného překlopení uvažuje rozdíl mezi mapováním a závodem. Během mapování ukládá čas do pomocné proměnné a uvažuje o každém překřížení jako o reálném s ohledem na to, že není nijak ovlivňováno externími vlivy. Oproti tomu v režimu závodu porovnává čas každého překřížení s překřížením naměřeným v první fázi a vyhodnocuje, jestli každé nové měření splňuje podmínku korektní délky trvání, která je určena procentuálně z referenčního času. Za předpokladu splnění podmínek je měření vyhodnoceno jako korektní, tudíž je překřížení bráno reálně. V opačném případě zbytek programu nepozná, že bylo překřížení detekováno.

Měřit referenční čas trvání průjezdu překřížením je vhodné během mapovacího módu, kdy jede auto konstantní rychlostí. Ve své podstatě nezáleží, které překříženi si vybereme pro měření, jelikož jsou totožná a průměrováním jednotlivých hodnot nedocílíme žádného zlepšení.

7.3 Porovnání projetého úseku

Auto může začít mapovat hned při první změně směru, kdy je zřejmé, že tuto první namapovanou část zachytí od počátku, protože mapovat hned od startu by mohlo narušit měření. Na startu nelze říci jestli jsme na začátku roviny nebo v jejím prostředku. Tento předpoklad platí tehdy, pokud je dráha mapována pouze z akcelerometru a ne za pomoci jiných čidel. V takových případech lze měřit od počátku a s lepší přesností určit start.

Vzhledem k tomu, že na trati může být více stejných úseků je nutné porovnávat vždy více po sobě jdoucích prvků. Úspěšnost pak závisí na množství analyzovaných dat. Lze porovnávat

trvání jednotlivých úseků, průměrné nebo maximální přetížení případně délku ale základem je správná posloupnost za sebou jdoucích směrů. Nicméně jednotlivá měření nejsou vždy stejná a mohou se mírně lišit. Otázkou také je, kdy začít samotné porovnávání. Počátečním impulzem může být druhé překřížení, které je umístěno za poměrně dlouhou částí projeté trati. Toto omezení nám sníží riziko nalezení úseků podobajících se počátečnímu.

7.4 Ukládání na kartu

Kromě LED diod připojených k DPS je zápis jedinou možností jak debugovat [7] zdrojový kód a zjistit nad čím procesor přemýšlí. Také se dá využít k ukládání nasnímaných dat sloužících pro tvorbu mapy a kromě vnitřní paměti mikroprocesoru je to jediná permanentní paměť, což lze využít k řešení problematiky výpadků auta z dráhy. Jedna z nejcennějších výhod je ukládání dat z akcelerometru pro analýzu během vývoje a jejich samotné grafické znázornění.

V programu je implementován souborový systém FAT [8], který usnadňuje budoucí čtení zaznamenaných informací v počítači. Také ale tvoří mezivrstvu mezi hardwarovou částí paměťové karty a uloženými daty, což nás nutí používat další funkce pro práci se souborovým systémem a proto dochází k prodlévám než se samotná data fyzicky uloží. Pro menší množství dat to není až tak závažný problém ale pro častější zápis je vhodné se obejít bez souborového systému a zapisovat data přímo bez další mezivrstvy.

Před otevřením souboru je nutné připojit souborový systém na kartě, aby samotné otevření bylo možné. Dále se pak k práci se souborem používají formátované vstupy a výstupy, stejně jako je tomu v jazyce C.(viz tab. 2/s. 30)

7.5 Bitová pole

Většina pomocných proměnných nese jednoduchou informaci *TRUE* nebo *FALSE* a k jejich reprezentaci stačí pouze jeden bit. Avšak minimální alokovatelná jednotka je byte a proto jsou pro optimalizaci a šetření paměti velice dobře využitelná bitová pole. Ve své podstatě se jedná o rozložení jednoho bytu na menší potřebné díly. V následující ukázce

[7] ladění tvořeného programu

[8] File Allocation Table je jednoduchý souborový systém původně určený pro MS DOS

Tabulka 2: Souhrn použitých funkcí pro práci s paměťovou kartou

Název funkce	Popis
f_mount()	Připojení souborového systému
f_open()	Otevření souboru
f_printf()	Formátovaný zápis do souboru
f_read()	Čtení ze souboru
f_eof()	Testování konce souboru
f_close()	Zavření souboru

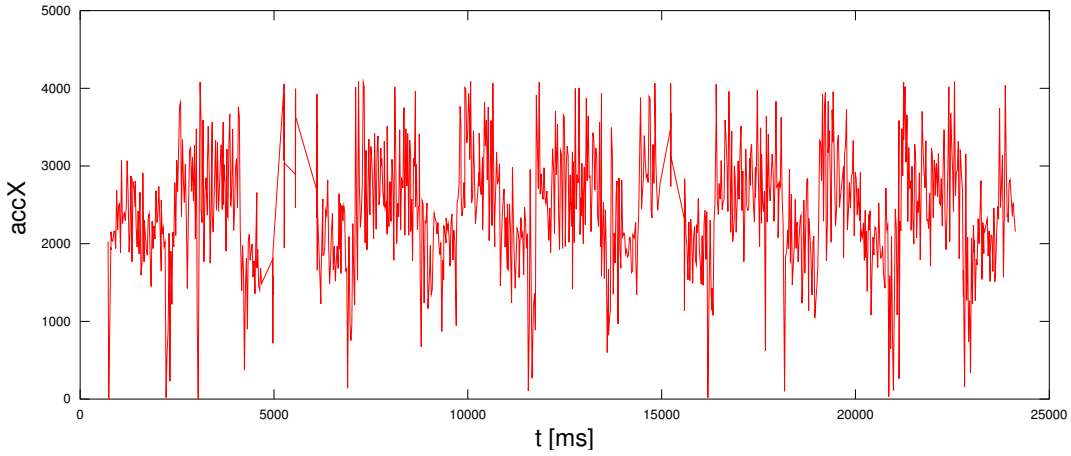
lze vidět definici ve zdrojovém kódu jazyka C, jeho alokaci a také zápis do této struktury. Ukázkový příklad zobrazuje práci s prvky mapy. Samotná definice není nikterak složitá, skládá se z nejmenšího datového typu, jména a za dvojtečkou je zapsána požadovaná velikost v bitech.

```
//definice struktury pro bitove pole
typedef struct{
  char leva : 1;
  char prava : 1;
  char rovina : 1;
}BitPOLE;
//alokace strukury
BitPOLE mapa;
//zapis do bitoveho pole
mapa.leva=1;
```

7.6 Plovoucí průměrovací filtr

Informace poskytnuté akcelerometrem jsou analogová data (viz obr. 3/s. 31) dle aktuálního přetížení působícího na auto a procesor je snímá pomocí AD převodníku
v 1/2 ms přerušeních.

Vzhledem k tomu, že naměřená data jsou nezpracovaná a mohou obsahovat



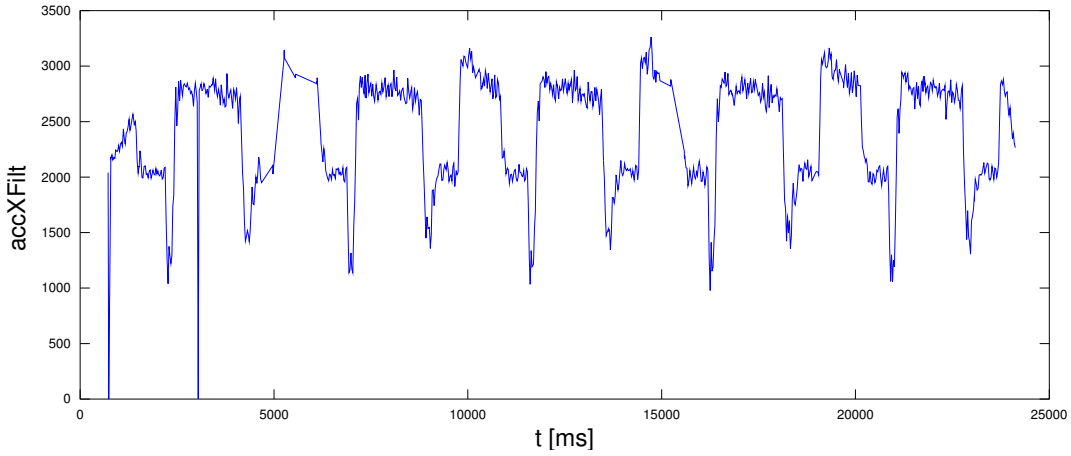
Obrázek 3: Analogová data z akcelerometru.

šum, je při měření analogových dat důležité použít filtr, který tento šum separuje od použitelných informací. Při použití surových dat by mohlo dojít ke špatné identifikaci zatáčky a polohy auta na dráze.

Kdybychom měli data nezávislá na čase, stačilo by použít obyčejný průměrovací filtr, který sumarizuje všechny získané hodnoty ale pro rychle jedoucí auto projíždějící zatáčkami není vhodný. V tomto případě přichází v úvahu plovoucí průměr tj. filtr, který průměruje posledních k prvků, kde počet průměrovaných prvků se nazývá plovoucí okno. Velikost tohoto plovoucího okna nám určuje kvalitu zpracovaných dat. Při velkém okně dochází k tomu, že auto detekuje zatáčku později, než by mělo z důvodu prodlevy změny průměru. Filtr musí nashromáždit značné množství nových prvků, aby se projevila změna stavu. Naopak malé okno hůře odděluje šum. Auto pak předčasně určuje zatáčky a může dojít i k chybě během měření. Jako optimální délku plovoucího okna lze použít 32 až 64 prvků (viz obr. 4/s. 32).

Během pozorovaní chování auta při použitém filtru o délce okna k=64, se jeho schopnost rozpoznávání zatáček výrazně zlepšila. Pro vizuální pozorování změny přetížení auta během jízdy pomůže nastavení předních reflektorů tak, aby jednotlivé LED diody odpovídaly dané zatáčce.

K optimalizaci filtru pomáhá nepoužití žádného cyklu tvořícího sumu. Rychlejší volba je pamatovat si první prvek plovoucího okna který odečteme od poslední sumy a poté



Obrázek 4: Data z akcelerometru po použití filtru.

přičteme nový. Takovéto řešení nepotřebuje k reprezentaci okna pole ale postačí již vypočtená suma. Ušetříme tím jednak čas výpočtu ale snížíme i nároky na paměť.

Místo plovoucího průměru lze použít Kalmanův filtr, který má základ v plovoucím průměrovacím filtru s tím, že je rozšířen o predikci chyby v měření. Používá se v případech, kdy nelze zaručit korektnost budoucích naměřených dat. V programu nebyla nutná implementace tohoto filtru.

7.7 Mapování dráhy

7.7.1 Určení směru

K určení směru slouží odfiltrovaná data z osy x poskytnuté akcelerometrem podle aktuálního přetížení. Během průjezdu rovinou se drží ve střední hladině a v zatáčkách hodnota přetížení klesá či stoupá podle směru zatáčky, kterou auto projíždí.

V programu jsou nastavené prahové hodnoty, kde při jejich překročení během porovnávání zjistíme, zda-li se auto nachází v levé nebo pravé zatáčce. Naopak pokud ani jedna z prahových hodnot nesplňuje podmínky zatáčky jede auto po rovině. Jejich správným nastavením určujeme citlivost auta a zamezujeme špatné, předčasné nebo pozdní detekci.

Nedílnou součástí správného měření je optimální rychlost. Ta ovlivňuje odstředivou sílu v zatáčkách působící na akcelerometr. Proto je důležité během mapování jet dostatečně

rychle, avšak tak, aby nedošlo k vypadnutí auta z dráhy. Naopak při pomalé jízdě dochází k detekci rovin v dlouhých zatáčkách, hlavně na vnějším okruhu, kde nepůsobí taková odstředivá síla.

Nastavení optimální rychlosti ovlivňuje adheze pneumatik. Správným odmaštěním zaručíme lepší přilnavost vozu k dráze a zabráníme prokluzu kol při vyšší rychlosti.

7.7.2 Rozpoznání nového směru

Přestože data z akcelerometru jsou snímána v 1/2 ms přerušeních a směr zjišťován každých 10 ms, je toto z dlouhodobého hlediska nevhodné, neboť je třeba zaznamenávat změnu směru. Změnu můžeme rozlišit pomocí proměnných, kde první nese informaci o aktuálním směru a druhá o předchozím. V momentě, kdy dojde k přechodu mezi stavy se proměnné liší a tudíž víme o novém směru auta. Ihned po tomto přechodu uložíme do proměnné předchozího směru proměnnou aktuální a auto dále čeká než dojde k detekci další změny.

Výchozí stav obou proměnných je rovina z důvodu startu auta na startovní rovince. Při změně směru tak dojde k okamžité aktualizaci a zamezíme tím zmatení programu, které by mohlo nastat v případě, že by program mapoval již od začátku.

Tento jednoduchý mechanizmus je využit během mapování a synchronizaci auta na dráze, viz. kapitola 9.1.

7.7.3 Mapa

Mapu tvoří jednotlivé naměřené úseky během mapování a ve zdrojovém kódu je reprezentována pomocí pole struktur. Mapa je také nositel užitečných informací získaných během mapovací jízdy pro nadcházející závodní režim.

Velikost pole je předem staticky dána, jelikož bez dodatečných funkcí nelze využít dynamickou alokaci jako je tomu během programovaní desktopových C aplikací pro operační systémy. Není to však přílišnou nevýhodou vzhledem k paměťovým možnostem procesoru, jehož paměť RAM má k dispozici 16 KByte prostoru, které nejsou zcela využity. Můžeme tedy velikost pole nadimenzovat tak, aby nedošlo k jeho přetečení během mapování.

K práci s mapou slouží několik proměnných, které určují její velikost, aktuální index zápisu a poslední index prvku mapy.

- Velikost mapy předdefinujeme na počátku programu a za její pomoci můžeme kontrovat program tak, aby nedošlo k zápisu do nepovolené části paměti, což by mohlo mít za následek fatální chyby.
- Aktuální index je z počátku odkaz na současnou pozici v mapě kde se ukládají prvky během její tvorby ale dále slouží ke čtení dat z aktuální pozice.
- Poslední prvek určíme během konce mapování a nadále slouží k ochraně proti čtení špatných informací z nezaplněné části mapy.

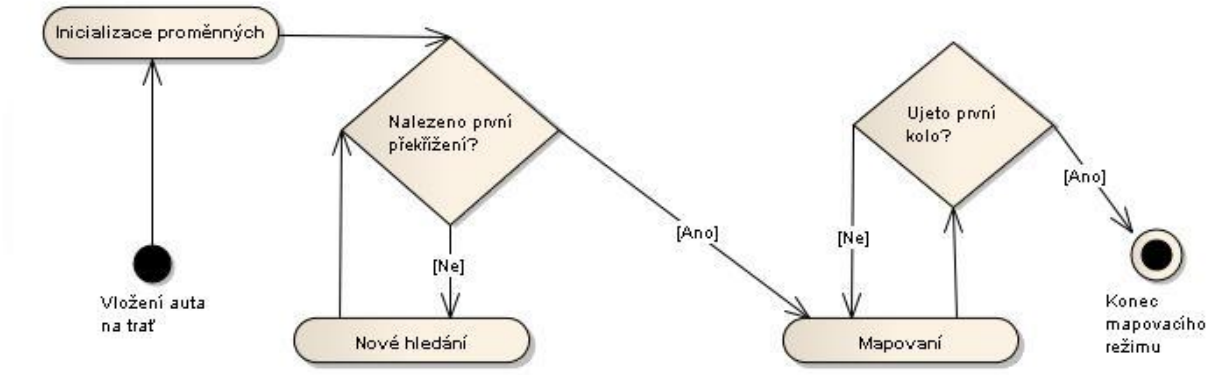
8 Mapovací režim

Mapovací režim, přestože je to malý kousek bloku, je velice důležitý pro budoucí závod. Vzhledem k tomu, že se auto později orientuje podle naměřených hodnot v tomto kole, je dobré této problematice věnovat více pozornosti.

Základem je mapovací rychlost, která určuje odstředivou sílu působící na akcelerometr. Čím vyšší rychlost, tím přesněji namapujeme přechod mezi směry ale nesmí být příliš vysoká aby auto během průjezdu všemi zatáčkami a překříženími nevykolejilo.

8.1 Tvorba mapy

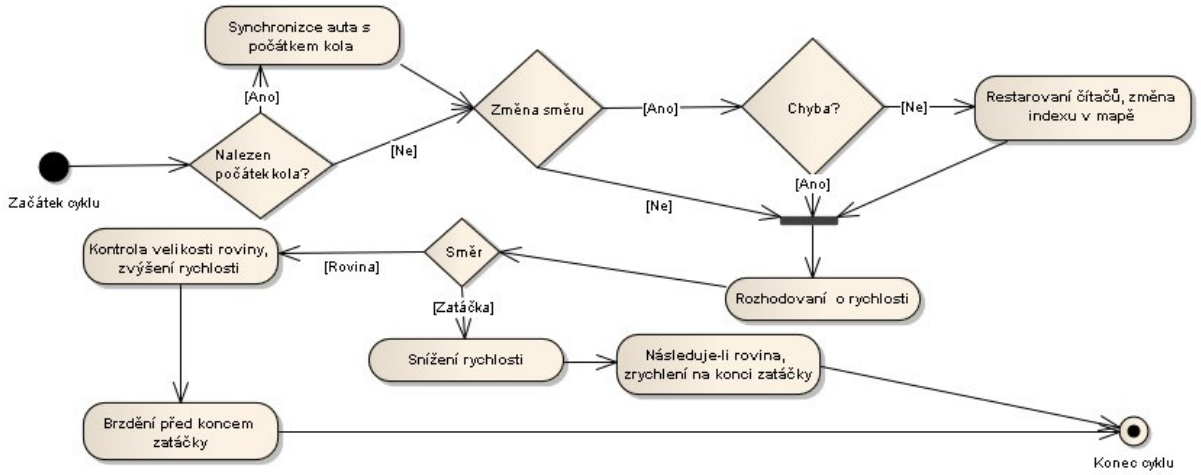
Samotná tvorba mapy začíná prvním projetím nejbližšího překřížení a končí jeho opakovaným průjezdem. V počátku jsou stejně jako při tvorbě nového prvku mapy resetovány potřebné proměnné např.: čas, vzdálenost,...



Obrázek 5: Graf popisující princip mapování.

9 Závodní režim

Během závodního režimu je nutné kalkulovat s jiným fyzickým chováním auta v důsledku akcelerace, decelerace a vyšších rychlostí při průjezdu zatáčkami. Celkově dochází ke smykům, prokluzu kol, vlnění zadní části vozu na vozovce nebo mikrovýpadkům proudu. Všechny tyto problémy zhoršují a znepřesňují měření a matou informacemi auto o jeho pozici na dráze. Je tedy nutné aplikovat na měřené hodnoty korekce a zmírnit tak jejich procento chybovosti.



Obrázek 6: Graf jednoho cyklu závodního režimu.

9.1 Synchronizace auta na dráze

Jako hlavní opěrný bod synchronizace slouží první nalezené překřížení. Tento způsob zůstal po celý vývoj programu využíván, pouze s rozdílem jiné implementace ve zdrojovém kódu pro další kombinace synchronizace vozidla na dráze.

Obecně během synchronizace dochází k restartování pomocných proměnných sloužících k uložení informací o autě a ukazatele do mapy.

9.1.1 Původní verze

Na počátku vývoje celého programu se auto synchronizovalo za pomocí naměřené vzdálenosti ze snímače otáček kola. Celkové měření vždy probíhalo od poslední synchronizace, kdy se čítač vzdálenosti resetoval na nulový stav.

Ukazatel do mapy se vázal s daty uloženými v mapě, tato data značila vzdálenost jednotlivých úseků od počátku kola. Pokud tedy ujetá vzdálenost byla delší než počátek následujícího prvku v mapě došlo k inkrementaci indexu.

Tato metoda se ukázala jako poměrně nespolehlivá a orientace auta na dráze byla velmi špatná. Díky technickým problémům snímače 6.1.3 auto pozdě rozpoznávalo zatáčky nebo naopak zpoždění v synchronizaci a přepnutí v mapě na prvek s rovinou až za ní samotnou. Ale hlavním problémem bylo předčasné přetečení mapy, v tomto případě auto udržovalo konstantní rychlost až do opětovné synchronizace s překřížením a tím pádem značné časové časové ztráty během závodu.

9.1.2 Současná verze

Poslední verze využívá dat z akcelerometru podle kterých dochází k synchronizaci. Auto využívá pro pohyb ukazatele v mapě základní změnu směru ale program opět musí počítat s jiným fyzickým chováním než při standardním mapování kde má auto konstantní rychlost. Špatná synchronizace probíhala při prudké akceleraci kdy se autu rozkolébala zadní náprava do stran. Jsou tedy zapotřebí jistá opatření, aby nedocházelo ke špatnému určení pozice modelu.

Nejjednodušší způsob odstranění problému je v momentě, kdy se změní směr nahlédnutím na následující prvek do mapy a pokud souhlasí lze inkrementovat ukazatel. V opačném případě se informativní proměnné změní na původní stav a opět se čeká na další změnu směru. Teoreticky i přes toto opatření může docházet ke zmatení polohy ale během praktické jízdy se tento jednoduchý mechanizmus ukázal jako spolehlivý.

Malé časové prodlevy se projevily při výjezdu ze zatáčky na rovinu, kdy program rozpoznal rovinu později než by měl. Celý problém tkví v datech z akcelerometru, která nejsou v daný moment díky vyšším rychlostem ještě ustálena. Dobrou odezvu by mohlo mít zkrácení plovoucího okna průměrovacího filtru během závodního režimu, kde by změna směru

měla na průměrovaných hodnotách náhlejší projev.

Další možností je vytvoření bufferu s plovoucími hodnotami načtenými z akcelerometru a jejich zprůměrováním procentuálně odhadnout aktuální směr. Bufferem by také šla určit velikost chyby pří výkyvu auta na dráze a podle určité pravděpodobnosti změnit ukazatel mapy na následující.

9.2 Určení rychlosti

Rychlost auta se řídí pulsně šířkovou modulací, generovanou z časovače mikroprocesoru a samotný časovač se nastavuje příslušným registrem, který ovlivňuje jeho cyklus. Tento namodulovaný signál je přiveden z výstupního pinu časovače na vstupní piny H-můstku.

Program auta zrychluje ve dvou různých situacích o kterých ví, že dráha pro zrychlení je dostatečně dlouhá nebo pokud po průjezdu úseku, ve kterém se nachází, následuje rovina. Pokud má prostor pro zrychlení, nastaví motoru maximální zvolenou rychlost aby získal čas.

V případě přicházející roviny program nahlédne do mapy, zjistí délku daného úseku a pokud tento plní podmínku minimální velikosti, nastaví vyšší předdefinovanou rychlost. V zatáčce opět čte informace z mapy ale s tím rozdílem, že se informuje o délce trvání daného úseku místo jeho délky a to z důvodu nespolehlivosti snímače otáček. Také nahlíží na další prvek v mapě zda-li má příznak roviny. Bez tohoto ověření by mohlo auto zrychlit v nevhodnou dobu a vylétnout z dráhy. Moment, kdy má zrychlit, pozná podle časového poměru mezi časem uloženým v mapě a podle aktuální hodnoty vnitřního časovače. Dále musí počítat s kratším časem potřebným pro průjezd zatáčkou během závodního režimu oproti času naměřeném v mapě při konstantní rychlosti z předchozího mapovacího režimu.

9.2.1 Statické

Statickým zrychlením je myšleno přivedení konkrétní hodnoty do registru časovače bez předchozího postupného navyšovaní. Není tedy potřeba žádné režie a postačuje pouhé uložení požadované hodnoty. Pokud nastavovaná rychlost není příliš vysoká, auto plynule zrychlí.

Tato metoda se ukázala být optimální a nedocházelo k nežádoucím vlivům, které by mohly ovlivnit závodní režim.

9.2.2 Dynamické

Pro dosažení efektivního zrychlení je možné použít postupné navyšovaní aktuální hodnoty určující rychlost auta. Do cílené rychlosti tak dynamicky auto akceleruje s větší razancí, jelikož kroky mezi jednotlivými rychlostmi jsou krátké a nemusí vyvíjet tolik energie na překonání hmotnosti auta.

Prudké zrychlení však může způsobit prokluz zadních kol, která poté nejsou v kontaktu s dráhou. Nedojdeme tedy k požadovanému výsledku ale naopak k časové ztrátě. Řešení toho problému můžeme najít v měření času mezi jednotlivými inkrementacemi.

9.3 Brzdění

Brzdění je velice důležitou součástí programu. Jeho využití umožňuje jízdu vyšší rychlostí po delší dobu.

Jede-li auto rovinou, porovnává vzdálenost z mapy s vzdáleností naměřenou snímačem otáček kol. Dokáže tak odhadnout vzdálenost před koncem zatáčky a určí kdy zpomalit, tedy pasivně zabrzdit předtím než vjede do zatáčky.

9.3.1 Pasivní

Snížení otáček motoru pasivně překonává, za pomoci magnetických sil uvnitř motoru, setrvačnost celého auta. Je to obdobná situace jako sundání nohy z plynového pedálu v osobním automobilu, kdy komprese motoru brzdí celé vozidlo bez použití brzd.

9.3.2 Aktivní

Aktivní brzdění je tvořeno přivedením signálu na druhý vstupní pin H-můstku, kdy dochází k rotaci motoru opačným směrem. Samotná účinnost brzdy se určuje rozdílem intenzit jednotlivých signálů na pinech H-můstku. Pokud jsou přivedené signály shodné, dochází k blokaci kol.

Tento způsob brzdění umožňuje efektivněji zpomalit auto na kratším úseku a tím pádem dovoluje vyšších rychlostí po delší dobu.

Velice důležité je mít správně nastavené příslušné registry tak, aby nedocházelo k protáčení kol v opačném směru vzhledem ke směru jízdy auta. Došlo by tak ke zpětné jízdě.

Závěr

Během práce jsem vyvinul program, který splnil podmínky zadání a umožnil autu optimální průjezd během závodu. Vývoj se ovšem neobešel bez jistých mechanických problémů, mezi které patří adheze pneumatik značně ovlivňující přilnavost auta. Řešení lze najít v pravidelném odmašťovaní technickým lihem. Další závada byla také na zničené čtečce karet poškozené při demontáži karoserie, čtečku bylo potřeba z náhradních dílů vyměnit.

Při zpracovávání této bakalářské práce jsem došel závěru, že před rozšířením hardwaru je potřeba zabývat se prvotními možnostmi, kterými auto disponuje. Jakýkoli vývoj dalších komponent může odvádět pozornost od práce s akcelerometrem, který při maximálním využití tvoří velice cenný zdroj informací.

Po zkušenostech získaných během této práce jsem dospěl k názoru, že hlavním cílem je vytvořit program co nejjednodušší a nejefektivnější. Každá zbytečnost navíc může způsobit větší chybovost celého algoritmu. Hardwarová rozšíření vyžadují velice pečlivé přípravy i precizní zpracování v programu. Jednotlivé opominuté drobnosti mohou vést k fatálním chybám.

Mohu potvrdit, že i takto jednoduchý program může docílit slušných výsledků, což dokazuje druhé místo ve školním kole na Technické Univerzitě v Liberci. Auto nezískalo první místo o 0,81 s a přišlo tak o možnost zúčastnit se velkého finále v Rožnově pod Radhoštěm, kde má společnost Freescale Semiconductor hlavní sídlo v ČR.

Na závěr bych rád poznamenal, že téma této práce bylo velice přínosné a rozšířilo mi obzory v této problematice. S prací tohoto typu jsem se setkal poprvé a přestože se vyskytly nějaké problémy popsané výše, povedlo se mi je úspěšně vyřešit.

Literatura

[1] ChaN: *ELM - FatFs Generic FAT File System Module.* [online], 2010, [cit. 2012-04-12]. URL: elm-chan.org/fsw/ff/00index_e.html

[2] Fuksa, Michal: *Pulzně šířková modulace.* [online], 2001, [cit. 2012-04-12]. URL: http://www.volny.cz/fuksam/povidani/pwm/pwm.htm

[3] Brejl, Milan: *Freescale Race Challenge.* [online], 2010, [cit. 2012-04-12]. URL: http://www.hw.cz/files/uploads/story/4314/frc2011-slotcarequipmentcz0.pdf

[4] Freescale Semiconductor, Inc.: *MCF51JM128 ColdFire® Integrated Microcontroller Reference Manual.* [online], 2009, [cit. 2012-04-12]. URL: http://www.freescale.com/files/32bit/doc/ref_manual/MCF51JM128RM.pdf

[5] Freescale Semiconductor, Inc.: *±1.5g, ±6g Three Axis Low–g Micromechined Accelerometer.* [online], 2008, [cit. 2012-04-12]. URL: http://www.freescale.com/files/sensors/doc/data_sheet/MMA7361L.pdf

[6] Freescale Semiconductor, Inc.: *5.0 A Throttle Control H-bridge.* [online], 2008, [cit. 2012-04-12]. URL: http://www.freescale.com/files/analog/doc/data_sheet/MC33931.pdf

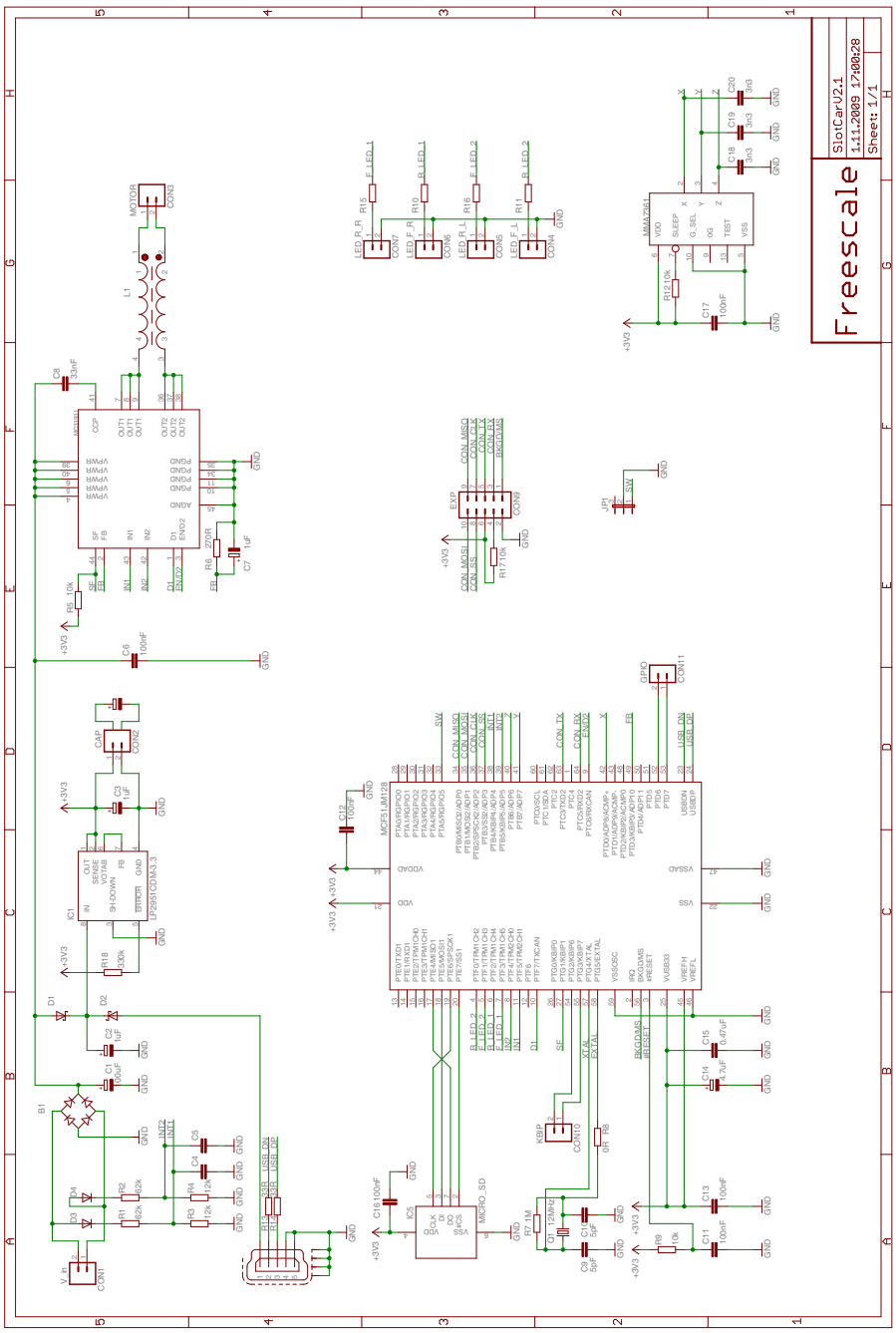
[7] Brejl, Milan: *Freescale Race Challenge 2012.* [online], 2011, [cit. 2012-04-12]. URL: http://www.hw.cz/teorie-a-praxe/mimochodem/freescale-race-challenge-2012-soutez-samoridicich-auticek-na-autodrahu

[8] Winkler, Zbyněk: *jak se vypořádat s nepřesnými daty.* [online], 2005, [cit. 2012-04-12]. URL: http://robotika.cz/guide/filtering/en

[9] Vít, Jakub: *Samořízené elektrické auto.* Bakalářská práce, NTI FM TUL, Liberec, 2011.

A Přílohy

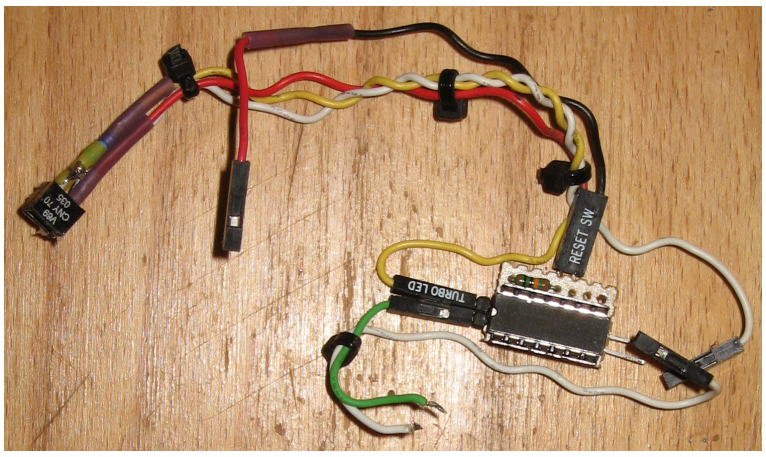
A.1 Schéma plošného spoje



Obrázek 7: Zapojení komponent.

A.2 Snímač otáček kola

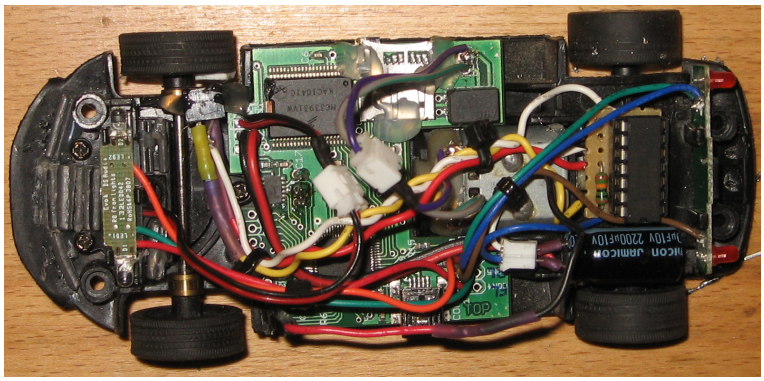
Obrázek 8: Reflexní plocha pro čidlo otáček.



Obrázek 9: Čidlo otáček kol.

A.3 Model Auta

Obrázek 10: Model auta během jízdy.



Obrázek 11: Pohled na vnitřní elektroniku.

A.4 Obsah adresářů na přiloženém CD

- **Baklářská práce:** Text bakalářské práce ve formátu .pdf.
- **Zdrojový kód:** Zdrojový kód v podobě projektu z prostředí CodeWarrior.
- **Obrázky:** Použitý obrazový materiál.